Ramesh

Decision Trees in Q

Version 0.1 March 22, 2018

Contents

Decision Trees in Q

1 Introduction

1.1 Notations

- Let $F = \{f_i\}$ be a table of F4 vectors, representing the features.
- Let g be a B1 vector, representing the outcome which we wish to predict

A decision tree is a Lua table where each element identifies

1. a feature
2. a threshold, the default comparison is always $\leq$.
3. a left decision tree
4. a right decision tree

Invariant 1 $forall f \in F, f : length() = g : length()$

Let α be minimum benefit required to continue branching
Initialize, $T = \{\}$
F, g as described above
function DT(T, F, g)
    $n_P, n = Q.sum(g)$
    forall $f \in F :\ s(f), b(f) = \text{Benefit}(f, g, n_N, n_P)$
    Let f' be feature with maximum benefit
    if benefit $\geq \alpha$ **then**
        $x = Q.vsgt(f', s(f'))$
        $n_R, n = Q.sum(x)$
        $F_L = F_R = \{\}$
        forall $f \in F$ **do**
            $Q.reorder(f, x)$
            $F_L = F_L \cup Q.vector(f, 0, n_L)$
            $F_R = F_L \cup Q.vector(f, n_L, n)$
        endfor
        T.feature = f'
        T.threshold = $b(f')$
        T.left = $\{\}$
        T.right = $\{\}$
        $DT(F_L, g_L, T_L)$
        $DT(F_R, g_R, T_R)$
    endif
end

Figure 1: Decision Tree algorithm

```
function Benefit(f, g, n_N, n_P)
    p_max = −∞
    b_opt = ⊥
    f′, g′ = Q.reorder(f, g)
    counter = ; counter[0] = 0; counter[1] = 0
    idx = 0
    n = f:length()
    REPEAT:
        b = f[idx]
        counter[g[idx]]++
        for ( j = idx; j < n; j++ ) do
            if f_j ≠ b then
                break
            endif
            counter[g[j]]++
        endfor
        p = WeightedBenefit(counter[0], counter[1], n_N, n_P)
        if p > p_max then
            p_max = p
            b_bot = b
        endif
        idx = j
        goto REPEAT
    DONE
end
```

Figure 2: Benefit Computation (numeric attributes)

function $\mathrm{WeightedBenefit}(n_N^L, n_P^L, n_N, n_P)$

$n_N^R = n_N - n_N^L$

$n_P^R = n_P - n_P^L$

$n_R = n_N^R + n_P^R$

$n_L = n_N^L + n_P^L$

return $\frac{n_L}{n} \times XXX + \frac{n_R}{n} \times YYY$

end

Figure 3: Weighted Benefit Computation

$$\boxed{\textbf{function}\ \ \mathrm{Benefit}(f, g, n_N, n_P)\ \textbf{end}}$$

Figure 4: Benefit Computation (boolean attributes)

$$\boxed{\textbf{function}\ \ \mathrm{Benefit}(f, g, n_N, n_P)\ \textbf{end}}$$

Figure 5: Benefit Computation (categorical attributes)